Beware of the leaked image list when using the TVS_CHECKBOXES style

devblogs.microsoft.com/oldnewthing/20171128-00

November 28, 2017

Raymond Chen

The `TVS_CHECKBOXES` tree view style is quirky, which is a nice way of saying that it's crazy.

As we noted last time, the support for check boxes was migrated in from external code, and it followed the pattern for external code. In particular, the state image list for the check boxes needs to be manually destroyed, because when you created the check boxes manually, you also needed to clean them up.

Yes, this goes against the general principle that things which were created by the control will be destroyed by the control. Like I said, the `TVS_CHECKBOXES` tree view style is quirky. And if you fail to accommodate this quirk, you end up with a resource leak.

MSDN suggests that you use the `TVM_GETIMAGELIST` message to retrieve the state image list, and then use `ImageList_Destroy` to destroy it. I prefer to exchange the image list out by setting the state image list to null, then destroying the returned image list (which is the previous image list). This avoids dangling references to a destroyed image list, and it also means that if somehow you try to clean up the image lists twice, the second one will simply not do anything since it won't see anything to clean up.

```
ImageList_Destroy(TreeView_SetImageList(hwndTV, nullptr, TVSIL_STATE));
```

We take advantage of the fact that the `HIMAGELIST` parameter to the `ImageList_Destroy` function is marked `_In_opt_`, which means that it is permissible to pass `nullptr`.

Okay, with these two common errors out of the way, I'll continue next time by beginning our exploration of tree view check boxes from the ground up.

Raymond Chen

Follow